Software Engineering Course Project

Restaurant Automation

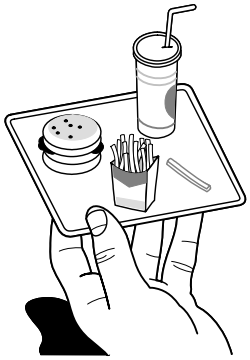
Project designed by Ivan Marsic

Department of Electrical and Computer Engineering

Rutgers University

Project website: http://www.ece.rutgers.edu/~marsic/books/SE/projects/

This project develops a computerized system to help restaurant personnel coordinate their activities and improve their services, and for the management to track business growth and create future plans. It is intended to be done by a team of 4-6 undergraduate students during an academic semester, in conjunction with lectures and other class activities. Other related projects and a software engineering textbook are available for download at this website:
http://www.ece.rutgers.edu/~marsic/books/SE/



1. Project Description

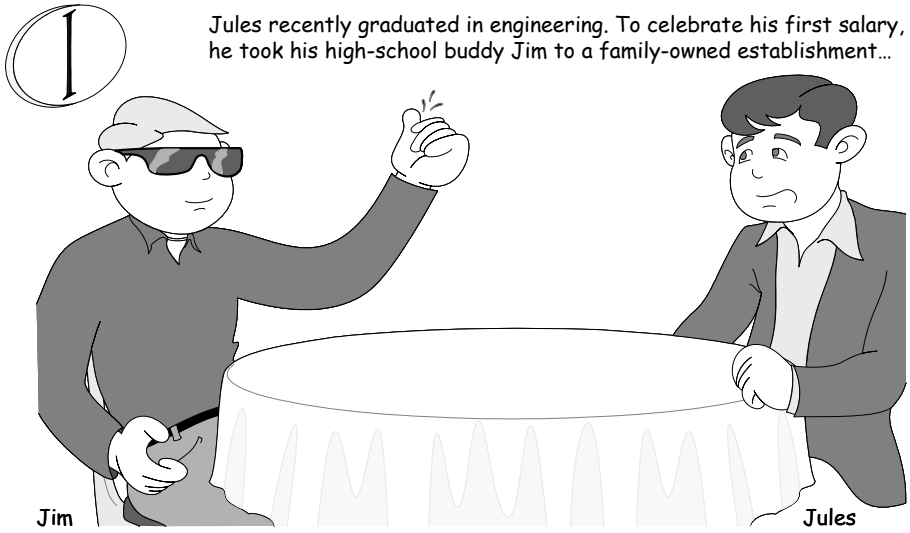
"Computers make it easier to do a lot of things, but most of the things they make it easier to do don't need to be done." —Andy Rooney

The goal for this project is to introduce automation in privately-owned restaurants, that is, small- to medium-sized establishments. Typical problems restaurant personnel are facing include:

- Coordination of their work activities
- Anticipating and handling periods of low/high patron traffic

- Recognizing trends early enough to take advantage of bestsellers or abandon the flops
- Lowering operating costs, and increasing efficiency/productivity and profits

Many restaurants are still operated using pen and paper methods, with little or no automation (see Figure 1). Patrons enter the facility to be greeted by a host, who often times has a "dry erase" diagram of the tables, maintained on a blackboard. The host can see the status of the tables based on whether or not they or someone else physically updates the diagram. Once seated a waiter tends to the costumers by jotting down the orders onto a piece of carbon paper and delivers it to the kitchen for proper food preparation. The waiter then has to periodically check back to find out when the meal is ready. When the food is done, the piece of carbon paper is saved for proper record keeping by the management. This "old fashion" system works but yields a large amount of tab receipts, wastes a lot of time and is simply out-of-date. In old fashion systems, waiters have to carry pads around to take orders, always have a working pen and be sure to keep each bill organized and "synchronized" with the proper table.



Another issue is record maintenance. In the old system, when everything is done by paper, the management is responsible to keep all information saved and organized, which is no easy task. Everyday tabs are collected, data needs to be organized and employees need to get paid. This requires a great deal of time and attention from the managers.

This project computerizes restaurant operation so that all information pertaining to patron's orders and staff activity will be conveniently shared and stored over the restaurant's intranet. Hosts will be able to view table status with a click of a button. The wait staff will be able to enter the patron's orders quickly and efficiently and then have it electronically delivered to the kitchen. The kitchen staff will be able to view the incoming orders and notify the proper wait staff when the food is ready. Bus boys will be able to view real-time floor status allowing them to know which tables are clean, dirty, or occupied. Most importantly, all of the restaurant information is organized and saved in the system database for the management viewing and archival. The analysis will consist of by-the-day and by-the-hour breakdowns of:

- Revenue and revenue percentage per menu item
- Menu item popularity

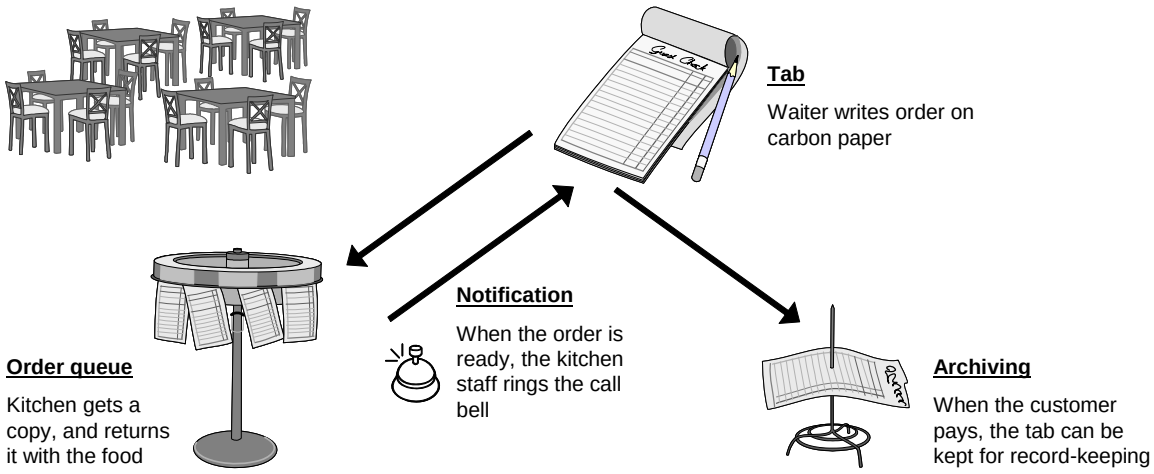


Figure 1: Old-fashioned restaurant operation.

- Personnel efficiency
- Average turnaround time (how long patrons spend in the restaurant)
- Average preparation time (time from when the order is placed to when it is ready)

There is no more abundance of papers and long hours of punching numbers. All data is automatically collected and processed allowing management to focus on analyzing the data rather than calculating it.

1.1 Statement of Requirements

By using a touch screen the restaurant staff can quickly and efficiently log in and complete the desired task. When a waiter logs in, they are greeted with a floor status screen in which their assigned tables are colored in. Their tables are colored according to status; green is open, yellow is occupied, red is dirty (see Figure 2). At this point a waiter can select a table to view its tab. Once a table is selected, the staff can choose from a number of options. If they select to add an item to the table's tab, they are presented with various categories of food items offered. Here they can select the appropriate category and then find the desired item. For example, if a patron ordered a Caesar salad, the waiter would login, select the table, and choose "Add Item." They would then select the "Soups/Salads" from the category list, and then select the desired salad from the items presented. They are then returned to that table's screen where they can choose to perform another task or logout. This saves the waiter from walking back and forth to the kitchen to deliver and check up on food orders. Orders placed by wait-staff using the computer terminals on the restaurant floor are displayed to the kitchen staff through a queue, i.e., on a first-in, first-out basis.

The supported employee roles are: *Host*, *Waiter*, *Cook*, *Busboy*, and *Manager*. Some of the direct links between some of the staff include: Host ↔ Waiter, Waiter ↔ Cook, and Busboy ↔ Host. Every user account in the system should have its own privileges. All the role-personalized home screens for each employee will be refreshed automatically when needed (when a table is marked ready; a table's order is prepared; a host assigns a waiter to a table; etc.). The Manager should

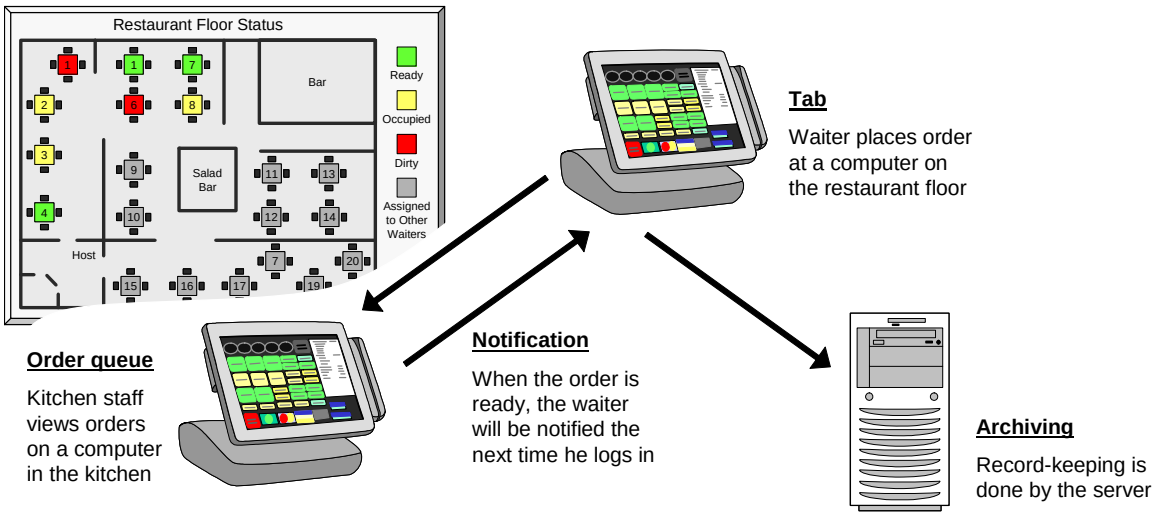


Figure 2: Restaurant automation facilitates staff coordination and record keeping.

have administrative power over employee profiles: the ability to create and modify profiles, track employee activities, and authorize restricted waiter activities. If the employee is a waiter, his/her profile also contains information about the tables for which he/she is responsible. From this profile, the individual tabs for those tables can be accessed. The manager should also have ability to manage other aspects of restaurant operations, such as inventory tracking and sale analysis.

There is an important choice of the type of computer terminals used by the waiters. All other personnel are either stationary, e.g., kitchen staff and hosts, or can access information at stationary terminals, e.g., busboys. If the waiters are required to use stationary terminals, they must memorize or jot down a specific table's order and then find an open computer, login and enter the information into the system. At this point everything else is done electronically. Another option is to have the waiters provided with handheld devices, which will be connected wirelessly to the rest of the system, completely eliminating the use of carbon paper.

There are certain advantages of stationary computer terminals over handheld devices: (*i*) a small number of fixed terminals can be time-shared by multiple personnel, so this solution may be cheaper and easier to maintain; (*ii*) they are fixed, so they cannot be lost or damaged in handling. Thus, in the first instantiation we will assume fixed terminals for the entire staff.

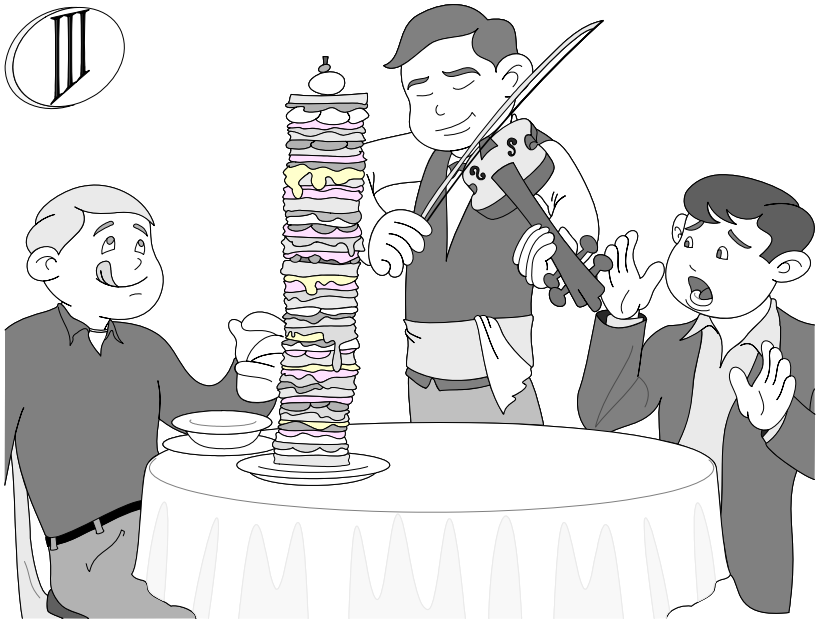
Throughout the restaurant there will be computer terminals for the staff to login. The system requires each user to login, complete their task and logout. Because this will require frequent logins and logouts, it may appear as an unnecessary overhead. With a limited number of terminals, staff will not be able to remain logged in because other employees need to use the computers. Logging in and out events can be exploited to trigger data updates and reorganization, as well as for delivering user-specific notifications of environment changes, such as to announce that the food is done or that a table needs to be cleaned. The only users who will be constantly logged in are the kitchen staff

and the host. They will be the only people using those terminals and therefore will not require frequent logouts.

Another design issue is that of how users identify themselves with the system. The considerable options are a touch screen, a card reader, or a typical keyboard. A touch screen allows users to carry less and use the system quickly and efficiently, although they need to memorize their login information. Another option is swipe cards, which would work by having the management assign each employee a swipe card. To make this system useful, a card reader would be needed to accompany every computer station, as illustrated in Figure 2. To make new cards for employees, the management would also need a card writer, as well as blank, non-programmed cards. The staff at many restaurants is constantly changing and this ongoing employee turnaround may lead to a considerable waste in time and money for making new cards. Our final option is to use a typical keyboard system. This would work the same as a touch screen but a keyboard would take up more space and be potentially slower. A touch screen serves the same purpose as a keyboard and allows for a smaller computer station. This is selected as the working solution.

Having to login frequently is annoying, particularly for cooks working with food. If you decide to have open access, then you may argue that certain restaurant areas are physically secure, so electronic security is less of a concern.

The final interface issue is specifying the floor plan of the restaurant and the table arrangement. Ideally, this feature should be included with the system to allow managers to alter the floor plan by moving, adding and removing tables. If the development team experiences lack of time in the course of the semester, an acceptable workaround should be sought. For example, a generic restaurant floor plan can be designed specifically for demo purposes. When a restaurant orders our software we will build and develop a floor plan for that specific establishment, making the software package unique to that establishment.



In addition to staff coordination, the system tracks everything electronically then organizes it. Employee hours are kept, allowing for rapid processing of the payroll. Revenue is tracked by day, week, or month. All this information is collected, saved, and then entered into table format for

easy reading by the management. The automatically generated statistics allow the management to see what portion of the revenue comes from what item, i.e., what are the most popular items. All this is done automatically and stays up to date with restaurant performance.

The developer(s) should count the number of clicks/keystrokes that are necessary to accomplish individual tasks. Make every effort to reduce the number of clicks/keystrokes in the system interaction, while not compromising functionality and security.

1.2 Domain Fieldwork

Visit local restaurant(s) and interview personnel to understand the current practice and identify the opportunities where introducing automation can help improve productivity and reduce errors.

It is a great advantage if some of the team members have experience working in restaurants. When writing your project reports, you should use your own experience, or that of others working in restaurants when describing and justifying your design decisions. In software engineering, personal familiarity with the problem domain is one of the greatest assets, and you should show it explicitly in your reports. (If you are currently working in a restaurant, then you should also interview your colleagues at work and quote their opinions in your reports.)

Perform an economic study to find the optimal kind of devices to be used by the restaurant personnel. On the one hand, waiters could have handheld PDAs to avoid re-typing the order. On the other hand, fewer fixed computers on the restaurant floor can be used by multiple waiters. Additionally, consider the possibility of loss and damage, particularly for handheld devices.

2. Extensions

If multiple teams select this project, different teams can focus on one of the following:

1. A graphics tool for managing the restaurant layout
2. A statistics processing suite for tracking the restaurant performance
3. A PDA-based user interface for waiters

A possible extension is to consider other roles, such as bartender or even customer. A customer interface could be installed on tables, so that customers are able to call the waiter through this individual interface to modify their order or request a bill when finished dining.

2.1 Supporting Customer Devices

An obvious target for automation is replacing the traditional pen and paper method of taking customer orders with electronic devices. One option is that the restaurant provides an electronic device for each table or customer. An example is a restaurant pager, known as “restaurant alert device” (RAD). When a customer walks in, they are greeted by a hostess and handed a RAD. The RAD is a two-way pager designed to notify the customer that their table is ready Once they are


Visit http://electronics.howstuffworks.com/everyday-tech/restaurant-pager.htm for information about restaurant pagers

seated, the hostess will link the RAD device to their table. The customer can then use RAD to hail the waiter at any time.

In the spirit of corporate BYOD (Bring Your Own Device) trends, we may consider allowing customers to use their own smartphones or tablet computers to order food. The customer would visit the restaurant website and order the food directly. A key problem is to know that the customer is actually in the restaurant and to associate the customer with the table at which they are seated. A potential solution is based on the fact that apps for scanning QR codes can be found on nearly all smartphone devices. We would use two cards with QR code (see http://en.wikipedia.org/wiki/QR_code) for each table. When a customer walks in, they are greeted by a hostess and handed one card with QR code. This card verifies that the customer is currently at the restaurant premises as was assigned a table. In addition, each table has a permanently affixed card with QR code. This second card identifies the table. The customer would use their smartphone to scan both QR codes and in this way prove that they are in the restaurant and provide their table information.

One may wonder if two QR-coded cards are needed or one is enough. If a QR code is present only on each table, it could be photographed and reused later from a remote location. One the other hand, if the hostess is to hand out a card with table-specific information, then it may be complicated for the hostess to quickly find the matching card. Instead, the QR code on the cards handed out by the hostess can be the same for all tables, but may need to be periodically changed to prevent misuse.

If the hostess' cards will all have the same QR code, we may simplify things even further so that no cards are handed out to customers. Each hostess would hold a single QR-coded card that is valid for a given period. When a customer is escorted by a hostess to a table and seated, the hostess simply shows this card to the customer to scan the QR code with their device. The hostess next asks the customer to scan the QR code of the table at which they are seated, which completes the authentication and identification process.

2.2 Other Extensions

An interesting issue is about the coordination of food ordering cooking and delivery. The order is usually taken initially for a full course dinner (soup, appetizer, entrée, and dessert). The issue is when the waiter should send order to the kitchen and when the cooks should start preparing the food. Ideally this should be coordinated so that the next course is not delivered before the customer is finished with the previous one or that customer waits for too long between the courses. It is not feasible to offload the timing decisions to cooks. The cooks must handle multiple orders simultaneously and precise timing may be difficult to manage. We could have the waiter to monitor and notify the cook when to start the next course. The waiter's user interface for this task could be simple: select the table number and press a button "next." An additional issue is that customers at the same table may order at different times, e.g., if someone arrive late.

Another extension of this system is to help cooks make decisions about what type of food to cook, how much to cook, and when to cook it. The conflicting demands on cooks are (*i*) to have enough food already prepared to meet customer demand in timely fashion, and (*ii*) not to prepare

too much food that will not be purchased and will go to waste. The following paper documents problems encountered by one such system in the past:

A. M. Bisantz, S. M. Cohen, and M. Gravelle, "To cook or not to cook: A case study of decision aiding in quick-service restaurant environments," Report No. GIT-CS-96/03, College of Computing, Georgia Institute of Technology, Atlanta, GA, 1996. Online at: http://www.eng.buffalo.edu/~bisantz/pubs/c_pap.html

Brendan McCorkle came up with an idea, called Texturant, which is software restaurants can use to alert diners via text message when tables are ready. This idea may be patented (contact Mr. McCorkle http://www.linkedin.com/pub/brendan-mccorkle/a/91a/ab5 for details), but I believe that it should be acceptable to implement it for an academic project.

An interesting problem is also tracking and comparing performance statistics for restaurants with multiple locations.

3. Additional Information

Additional information about this project can be found at the project website, http://www.ece.rutgers.edu/~marsic/books/SE/projects/. Also, project reports along with running software developed by students at Rutgers University are available for download at this website. See also Problem 2.6 at the end of Chapter 2 of the accompanying software engineering book, the solution of which can be found at the back of the book.